Forward Propagation in a Recurrent Neural Network (RNN)

Overview

This document details the mathematical equations used in forward propagation for a Recurrent Neural Network (RNN). Forward propagation is the process of computing the hidden states and outputs of the RNN for a given sequence of inputs.

Mathematical Steps in Forward Propagation

1. Input Transformation

At each time step t, the input vector x_t is combined with the previous hidden state h_{t-1} to compute the pre-activation value z_t for the hidden state:

$$z_t = W_x x_t + W_h h_{t-1} + b_h$$

where:

- $W_x \in \mathbb{R}^{\text{hidden\_size} \times \text{input\_size}}$: Weight matrix for the input-to-hidden connection.
- $W_h \in \mathbb{R}^{\text{hidden\_size} \times \text{hidden\_size}}$: Weight matrix for the hidden-to-hidden connection.
- $b_h \in \mathbb{R}^{\text{hidden\_size}}$: Bias vector for the hidden layer.
- $x_t \in \mathbb{R}^{\text{input\_size}}$: Input vector at time step t.
- $h_{t-1} \in \mathbb{R}^{\text{hidden\_size}}$: Hidden state vector from the previous time step.

2. Hidden State Activation

The hidden state h_t is updated by applying a non-linear activation function f to the pre-activation value z_t:

$$h_t = f(z_t)$$

where f is typically a non-linear activation function such as:

- ReLU: $f(z) = \max(0, z)$
- Sigmoid: $f(z) = \frac{1}{1+e^{-z}}$
- Tanh: $f(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$

3. Output Computation

The output at each time step t is computed using the hidden state h_t:

$$y_t = W_y h_t + b_y$$

where:

- $W_y \in \mathbb{R}^{\text{output\_size} \times \text{hidden\_size}}$: Weight matrix for the hidden-to-output connection.
- $b_y \in \mathbb{R}^{\text{output\_size}}$: Bias vector for the output layer.

4. Output Activation

The raw output y_t is transformed using an activation function g to produce the final output $\hat{y}_t$:

$$\hat{y}_t = g(y_t)$$

The choice of g depends on the task:

- For regression tasks, $g(y_t)$ is often the identity function.
- For classification tasks, $g(y_t)$ is often the softmax function:

$$\text{Softmax}(y_t)_i = \frac{e^{y_{t,i}}}{\sum_j e^{y_{t,j}}}$$

  where $y_{t,i}$ is the i-th element of y_t.

Summary of Forward Propagation

For a sequence of inputs $\{x_1, x_2, \ldots, x_T\}$, forward propagation proceeds as follows:

1. Initialize the hidden state h_0 (often set to a vector of zeros).
2. For each time step $t = 1, \ldots, T$:
   (a) Compute the pre-activation z_t:
   $$z_t = W_x x_t + W_h h_{t-1} + b_h$$
   (b) Update the hidden state h_t:
   $$h_t = f(z_t)$$
   (c) Compute the raw output y_t:
   $$y_t = W_y h_t + b_y$$
   (d) Apply the output activation to compute $\hat{y}_t$:
   $$\hat{y}_t = g(y_t)$$

Conclusion

Forward propagation in an RNN involves iteratively computing hidden states and outputs for each time step. The equations described in this document form the mathematical foundation for processing sequential data in an RNN.